Les enregistrements

Rappel

- **Les tableaux à une dimension**
- Un tableau à une dimension peut être vu comme une liste d'éléments de même type. On le représente souvent comme une suite de cases contenant chacune une valeur.

- Un tableau à une dimension :
  - ❖ Possède **un nom**
  - ❖ Possède un **nombre d'éléments** ou de **cases** qui représente sa **taille**
  - ❖ Occupe une zone mémoire contigüe.
  - ❖ Possède une taille **finie** et **fixe**.

Rappel

- Les cases du tableau :
  - Chaque case du tableau est appelée **élément.**
  - Tous les éléments d'un tableau sont du **même type.**
- La déclaration d’un tableau de variables comporte 3 éléments :
  - Le nom du tableau, car toute variable possède un nom
  - La taille du tableau, qui correspond au nombre de cases du tableau
  - Le type des données contenues dans le tableau

Var nomTableau **: tableau [1.. taille] de typeDonnées ;**

- Pour désigner un élément du tableau, on indique le nom du tableau suivi de l'indice de l'élément (T[i])

Introduction

Problème:

- On veut afficher les moyennes des notes de TP et d’examen obtenues par les étudiants d'une classe du module de programmation.
- Pour cela, on souhaite également mémoriser le nom et le prénom de chaque étudiant.

Comment mémoriser toutes ces informations afin de les manipuler ?

Introduction

Solution 1: Utiliser plusieurs tableaux

const

nb_etud = 20;

var

Nom : **tableau** [1 .. nb_etud] de chaîne de caractères;

Prénom : **tableau** [1 .. nb_etud] de chaîne de caractères;

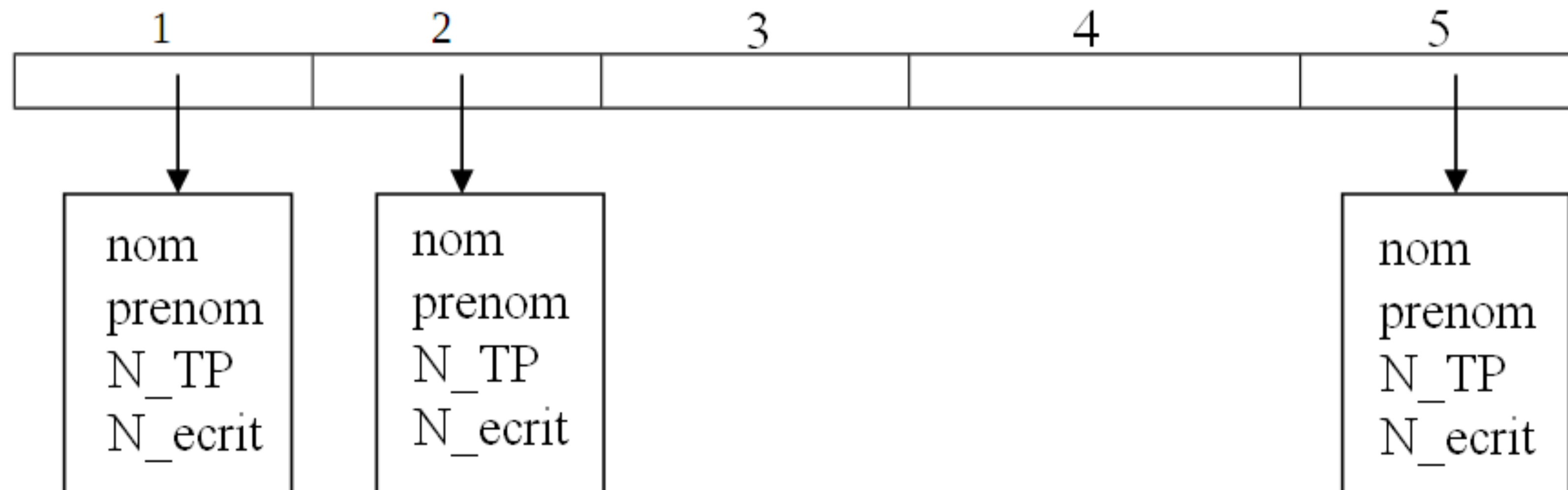
TP : **tableau** [1 .. nb_etud] de réel;

Examen : **tableau** [1 .. nb_etud] de réel;

Introduction

Nom	BOUMEDIENNE	BOUDIAF	MOKHTARI	……………..	AYED
Prénom	AEK	MOHAMED	ALI	……………..	WALID
TP	11	10	15	……………..	13,5
Examen	09	10	12	……………..	15

Inconvénient : manipulation de plusieurs tableaux



Introduction

- Solution 2 : Utiliser un tableau à deux dimensions

1	**2**	**…**	**nb_eleve**
nom	nom	…	nom
prénom	prénom	…	prénom
TP	TP	…	TP
Examen	Examen	…	Examen

Pb: Tous les éléments d'un tableau doivent être du **même type.**

Les enregistrements

1. **Définition:**

Un enregistrement est un type de données structurées, il permet de regrouper dans une même structure des données de différents types.

1. **Caractéristique:**

- Dans un tableau, tous les constituants doivent être semblables. Ce n'est pas le cas des enregistrements, qui sont des variables composées de plusieurs variables (ou **champs**) de types différents.
- Dans un enregistrement, chaque champ doit avoir un nom différent. Mais pour des enregistrements différents on peut réutiliser le même nom de champ.

Déclaration d’un enregistrement

Déclaration d'un **type** structuré enregistrement :

- Avant d’utiliser un enregistrement, il est tout d’abord nécessaire de déclarer un type enregistrement
- Un type enregistrement se définit dans un bloc type de la manière suivante :

```
Type
     nom_enregistrement  =  ENREGISTREMENT
            Champ 1 : type1;
            Champ 2 : type2;
            ...
            Champ n : type n ;

      FIN;
```

Les types des variables `Champ 1, Champ 2,..., Champ n` peuvent être de différents types

Une fois le type enregistrement définit, une variable de ce type se déclare dans un bloc **var**.

Dans notre exemple, nous aurions écrit :

```
Var     Nom_de_var : nom_enregistrement ;
```

Définition d'une variable de type enregistrement

- La méthode est donc la même que pour définir n'importe quelle variable, par exemple :

```
TYPE
      Etudiant = ENREGISTREMENT
            Nom, prénom : chaîne ;
            Note_TP, Note_examen : réel ;
      FIN ;
Var
      Etud1, Etud2 : Etudiant;
```

Tab : Tableau [1..400] de etudiant ;

Représentation : les enregistrements sont composés de plusieurs zones de données, correspondant aux champs :

	Etud1.nom	Etud1.prénom	Etud1.N_TP	Etud1.N_examen
Etud 1				
	Etud2.nom	Etud2.prénom	Etud2.N_TP	Etud2.N_exmen
Etud 2				

Manipulation d’un enregistrement

- La manipulation d'un enregistrement se fait au travers de ses champs.
- Comme pour les tableaux, il n'est pas possible de manipuler un enregistrement globalement, sauf pour **affecter un enregistrement à un autre** de même type.
- Par exemple, pour afficher un enregistrement il faut afficher tous ses champs.
- **Accès aux champs d'un enregistrement :**
- Pour référencer ou accéder a un champ quelconque d'un enregistrement, on utilise la notation pointée.
- La notation pointée utilise la variable enregistrement (identificateur) suivie d’un point puis du nom du champ auquel on veut accéder. Soit :

Exemple 1:

```
VAR
        Etud : Etudiant ;
```

- Pour accéder au nom de l'étudiant : `Etud.nom`
- Pour accéder à la note TP : `Etud.N_TP`

Lecture/ Écriture

- La lecture et l'écriture des valeurs des différents champs d'une variable enregistrement, via l'instruction ***lire***, se fait comme suit :

```
Lire (Etud.nom) ;
Lire (Etud.prénom) ;
Lire (Etud.N_TP) ;
Lire (Etud.N_examen) ;
Lire (Etud.nom, Etud.prénom, Etud.N_TP,
Etud.N_examen) ;
```

```
Écrire (Etud.nom) ;
Écrire ("prénom : ", Etud.prénom) ;
Écrire (Etud.N_TP) ;
Écrire (Etud.N_écrit) ;
```

Avec etud faire
Lire (nom, Pren, NTP..);
Favec;

Exemple : Programme qui affiche la différence d'âge entre ces deux personnes.

```
Type
        personne = enregistrement
                nom : chaîne;
                prénom : chaîne;
                age : entier;
        FinStructure;
Var
        pr1, pr2: personne;
Début
        Ecrire ("Entrez le nom puis l'age de l'apprenant 1");
        lire (pr1.nom, pr1.age); // il est impossible d'écrire Saisir pers1;
        ecrire ("Entrez le nom puis l'âge de l'apprenant 2");
        lire (pr2.nom, pr2.age);
        ecrire ("La différence d'âge entre ", pr1.nom, " et ", pr2.nom, " est de ");
        Si (pr1.age > pr2.age)Alors
                Aff pr1.age - pr2.age, " ans ";
        Sinon
                Aff pr2.age - pr1.age, " ans ";
        FinSi;
Fin.
```

Affectation

L'affectation de valeurs aux différents champs d'une variable de type enregistrement se fait par une opération de type :

variable.champ ← valeur;

- **Exemple :**
  - `Etud.nom ← "BOUMEDIENNE" ;`
  - `Etud.prénom ← "Mohamed" ;`
  - `Etud.N_TP ← 15 ;`
  - `Etud.N_examen ← 13.25 ;`
- Comme l'affectation fonctionne toujours pour des variables de même type, Il est possible d’utiliser l’opérateur d’affectation entre deux enregistrements de même type.

```
nom_enregistrement1 ← nom_enregistrement2 ;
```

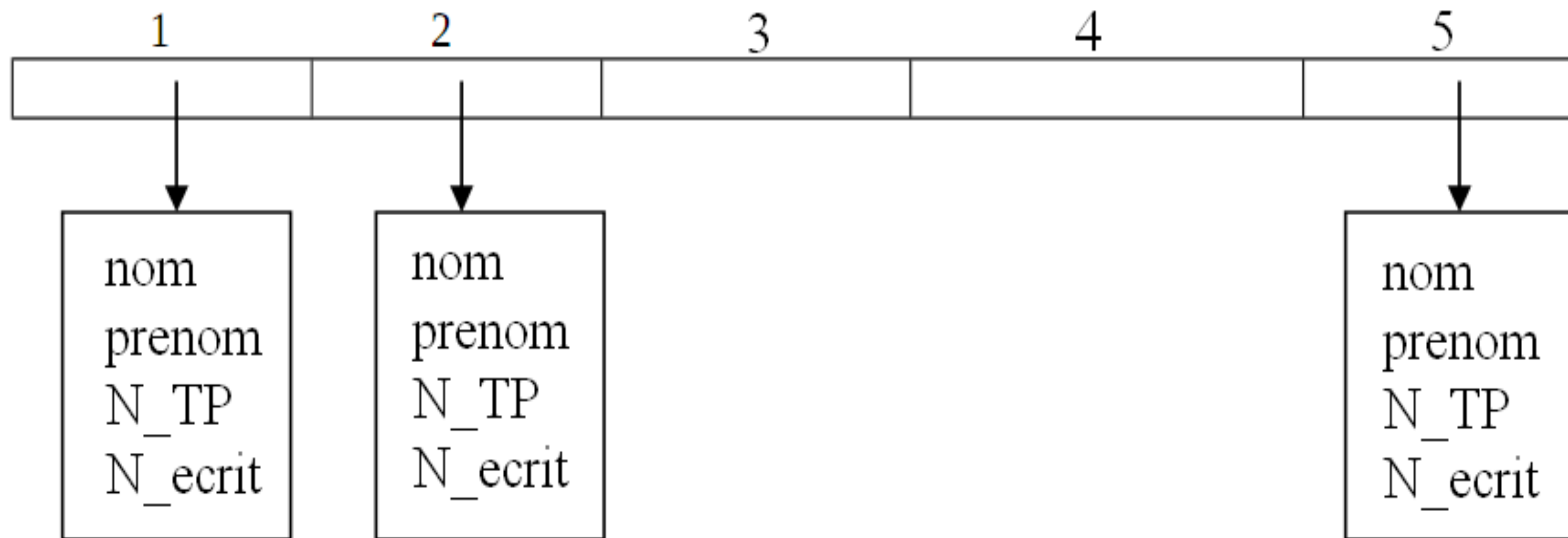
Tableaux d’enregistrement

- **4.5) Tableau d'enregistrements :**
- Il est possible de créer un tableau regroupant tout les variables de type enregistrement (exemple: les étudiants du groupe). Il s’agit alors d’un tableau d’enregistrements, (Exemple: ***`tabEtudiant`***).
- Un tableau ne peut grouper ou contenir que des éléments de même type, et puisque les éléments d'un enregistrement sont du même type qui est celui de l'enregistrement, on peut utiliser un tableau ou un vecteur d'enregistrements.
- La syntaxe de déclaration de la variable *`tabEtudiant`* utilisant les enregistrements des étudiants est :

```
Var
      Tabetudiant : tableau [1…5] de Etudiant ;
```

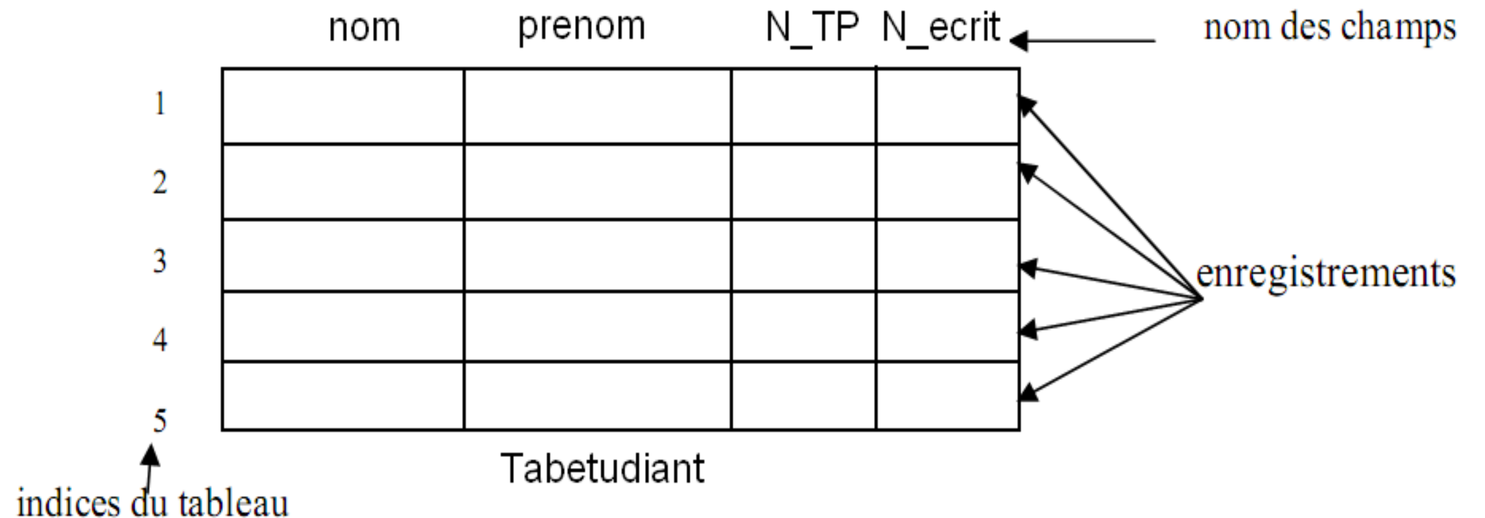
Tableaux d'enregistrement

- Chaque élément du tableau est un enregistrement, contenant plusieurs variables de type différent.



Tableaux d'enregistrement

- On accède à un enregistrement par son indice dans le tableau.



- **Tabetudiant [2]** représente le deuxième étudiant du groupe
- **Tabetudiant [2].nom** représente le nom de deuxième étudiant du groupe

Exemple

- Lecture et affichage d'un tableau d'enregistrement

```
Algorithme  LireEcrire
  Type Individu = Enregistrement
      Nom, Prénom : chaine
      Age : entier
      Ville, Fonction : Chaine
  Fin

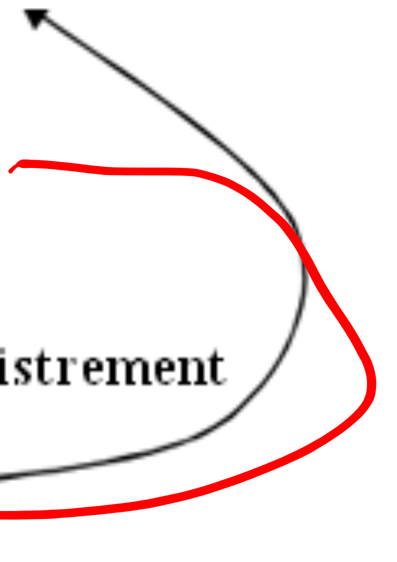
  Variables
      T : Tableau [1..1000] de Individu
      i,N : entier
Début
      Lire (N) { N est le nombre d'individus }

      Pour  i = 1 à N Faire
           Lire(T[i]. Nom)
           Lire(T[i]. Prénom)
           Lire(T[i]. Age)
           Lire(T[i]. Ville)
           Lire(T[i]. Fonction)
      FinPour

      Pour  I = 1 à N Faire
          Ecrire(T[I]. Nom)
          Ecrire(T[I]. Prénom)
          Ecrire(T[I]. Age)
          Ecrire(T[I]. Ville)
          Ecrire(T[I]. Fonction)
      Fin Pour
FIN
```

Imbrication d'enregistrements

- Supposons que dans un enregistrement personne, nous ne voulions plus l'âge de la personne, mais sa date de naissance.
- Une date est composée de trois variables (jour, mois, année) indissociables.
- Une date correspond donc à une entité du monde réel qu'on doit représenter par un type enregistrement à 3 champs.
- Si on déclare le **type date** au préalable, on peut l'utiliser dans la déclaration du type personne pour le type de la date de naissance.



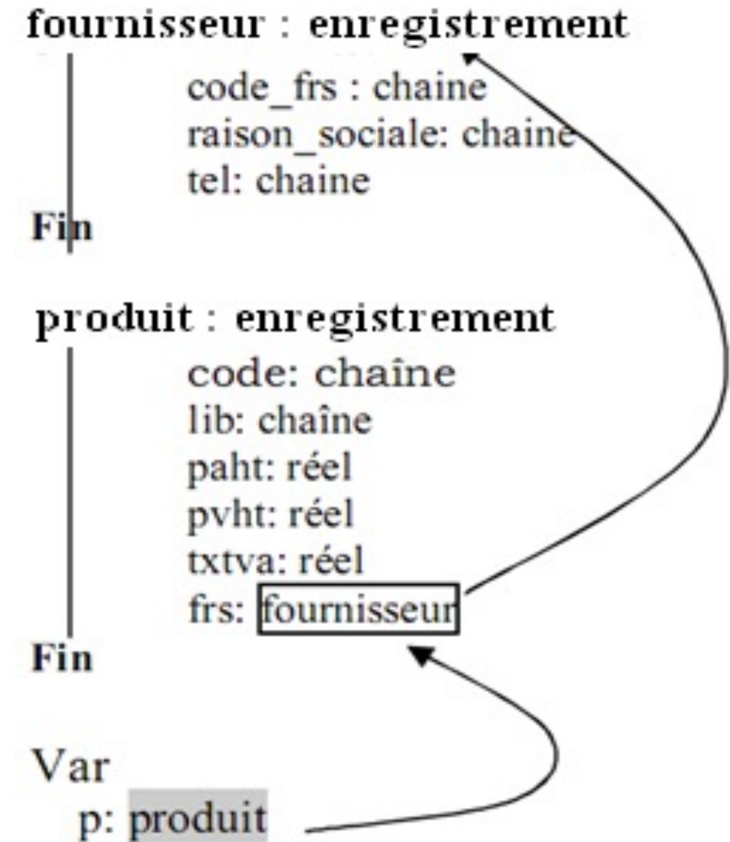
Imbrication d'enregistrements

- Pour accéder à l'année de naissance d'une personne, il faut utiliser deux fois l'opérateur '.'

```
pers1.ddn.année
```

- **Exemple**
- Un produit est livré par un seul fournisseur. Un fournisseur est caractérisé par son code, sa raison sociale et son numéro de téléphone.
- L'instruction qui permet d'afficher le numéro de téléphone du fournisseur du produit

```
ecrire ("téléphone du fournisseur est : ",
p.frs.tel)
```



Structure « Avec .. Faire »

n'existe pas en C.

- L'instruction **Avec ... Faire** (With .. Do) permet d'éviter les répétitions de la variable Enregistrement lors de l'accès à ses différents champs. Sa syntaxe générale est:

```
Avec variable Faire
    {ensemble d'actions}
Fin Avec
```

Cette structure s'utilise aussi bien avec une opération d'affectation, de lecture ou d'écriture.

Exemple

```
Avec Etud Faire
    prenom  ← "Mohamed" ;
    Ecrire ("Entrer la note de TP de l'étudiant : ") ;
    Lire (N_TP);
    Ecrire (« la note de l'examen est : ", N_examen);
Fin Avec
```